CONFIGURATION MANAGEMENT IN DEVOPS

To truly grasp the DevOps concept within an organization, you should understand a handful of essential primary disciplines. These include coding, building, testing, packaging, release, configuration, and monitoring. Each discipline comes with its own set of best practices.

Configuration management is important in DevOps because it helps you automate otherwise tedious tasks and allows an organization to increase agility. Moreover, configuration management supports DevOps holistically, and it's widely agreed upon that configuration management is essential to DevOps, as opposed to being just another component of the process.

In this article we look at configuration management, highlighting the following:

- What is configuration management?
- The three components that make-up DevOps comprehensive configuration management
- Outcomes of successful configuration management

If you're going through a DevOps transition in your enterprise business, you won't want to miss this guide on configuration management:

What is Configuration Management?

Configuration management occurs when a configuration platform is used to automate, monitor, design and manage otherwise manual configuration processes. System-wide changes take place across servers and networks, storage, applications, and other managed systems.

An important function of configuration management is defining the state of each system. By orchestrating these processes with a platform, organizations can ensure consistency across

integrated systems and increase efficiency. The result is that businesses can scale more readily without hiring additional IT management staff. Companies that otherwise wouldn't have the resources can grow by deploying a DevOps approach.

Configuration management is closely associated with change management, and as a result, the two terms are sometimes confused. Configuration management is most readily described as the automation, management, and maintenance of configurations at each state, while change management is the process by which configurations are redefined and changed to meet the conditions of new needs and dynamic circumstances.

A number of tools are available for those seeking to implement configuration management in their organizations. Puppet has carried the torch in pioneering configuration management, but other companies like Chef and Red Hat also offer intriguing suites of products to enhance configuration management processes. Proper configuration management is at the core of continuous testing and delivery, two key benefits of DevOps.

Components: Configuration Management in DevOps

Based on what we've discussed, you may have already gleaned that configuration management takes on the primary responsibility for three broad categories required for DevOps transformation: identification, control, and audit processes.

Identification:
The process of finding and cataloging system-wide configuration needs.

Control:
During configuration control, we see the importance of change management at work. It's highly likely that configuration needs will change over time, and configuration control allows this to happen in a controlled way as to not destabilize integrations and existing infrastructure.

Audit:
Like most audit processes, a configuration audit is a review of the existing systems to ensure that it stands up to compliance regulation and validations.

Like DevOps, configuration management is spread across both operational and development buckets within an organization. This is by design. There are primary components that go into the comprehensive configuration management required for DevOps:

- Artifact repository
- Source code repository
- Configuration management data architecture

Artifact Repository

An artifact repository is meant to store machine files. This can include binaries, test data, and libraries. Effectively, it's a database for files that people don't generally use. In DevOps, artifacts, like binaries, are a natural result of continuous integration. DevOps developers are always pushing out builds which, in turn, create artifact files that need to be stored, but not necessarily accessed.

Source Code Repository

Conversely, the source code repository is a database of source code which developers use. This database serves as a container for all the working code. Source code aside, it stores a number of useful components including various scripts and configuration files.

While some developers store binaries in this same repository, that's not a best practice. In DevOps, due to the sheer number of builds and off-shoot binaries, it's recommended that an artifact repository is developed for the purpose of storing binaries and other artifacts.

It's not hard to determine what goes into the source code repository. A quick litmus test is to ask yourself, "are the files human-readable?"

If yes, there's a good chance they belong in the source code repository as opposed to anywhere else. There are two types of source code repositories: centralized version control system (CVCS) and distributed version control system (DVCS).

In a CVCS, the source code lives in a centralized place, where it can be retrieved and stored. However, in DVCS, the code exists across multiple terminals useful in the development process. It's faster and more reliable. Most often, DVCS is the chosen source code repository of today's DevOps professionals.

Configuration Management Data Architecture

The idea of having data architecture dedicated to configuration management is a principle of ITIL service management framework. A configuration management database or (CMDB) is a relational database that spans across multiple systems and applications related to configuration management, including services, servers, applications, and databases to name a few.

CMDB is helpful for change management, as it allows users to audit the relationships between integrated systems before configuration changes are made. It's also a useful tool for provisioning as you can glean all identifying information for objects like servers. A CMBD is an essential tool when it comes to incident management, too, as it helps teams escalate issues to resolution.

Outcomes of Properly Managed Configurations

When a system is properly configured and managed, you can expect certain outcomes. Among these outcomes are delivering infrastructure-as-a-code and configuration-as-a-code. Below, we will look at the role each outcome has in configuration management:

Infrastructure-as-a-Code

Infrastructure-as-a-code (IaaC) in simplest terms is a code or script the automates the environment necessary for development, without manually completing all the steps necessary to build the environment. When we use the word 'environment' in this way, we are referring to the set up of all computing resources required to create the infrastructure to perform DevOps actions. This could be servers, networks of configurations and other resources.

Configuration-as-a-Code

As the name suggests, configuration-as-a-code (CaaC) is a string of code or script that standardizes configurations within a given resource, like a server or network. These configurations are applied during the deployment phase to ensure the configuration of the infrastructure makes sense for the application.

Benefits of IaaC and CaaC

Fans of continuous integration should be pretty familiar with the benefits of IaaC and CaaC, but for those new to DevOps, you'll want to know what to expect. These are some of the benefits of the two key outcomes defined in this section:

- Automation of the infrastructure environment provides standardization
- Setups are free of human error
- Collaboration is enhanced between operations and development
- Keeps configurations from drifting
- Makes infrastructure more flexible, ready to scale
- Each step is consistent across all resources
- Version control is a given

With these benefits applied to an organization, efficiency and greater agility are a natural result. DevOps is practically synonymous with configuration management, IaaC, and CaaC.

Configuration Management is DevOps

Seeing configuration management as separate and aside from DevOps creates a faulty perspective. Comprehensive configuration management, as described in this article, is essential to a properly functioning DevOps organization, as it lays the groundwork for far more automation than that which it impacts directly. With the inclusion of IaaC and CaaC in the development environment, enterprise businesses communicate better and function as a more agile development unit focused on continuous integration and continuous delivery.

The right tools are essential for configuration management. There are a number of heavy hitters throwing their hats in the ring to help your organization with configuration management. BMC is your essential resource for DevOps education and consultation, and we can help you integrate the right elements into your DevOps infrastructure.

When it comes to configuration management, BMC is a trusted partner. For more information on solutions that could elevate your enterprise organization, contact us today.